(19)日本国特許庁（ＪＰ）　(12)公開特許公報(Ａ)　(11)特許出願公開番号

特開平6－83598

(43)公開日　平成６年(1994)３月25日

(51)Int.Cl.[5]	識別記号	庁内整理番号	Ｆ Ｉ	技術表示箇所
Ｇ０６Ｆ　9/06	４３０　Ｅ	9367－5B		

審査請求　未請求　請求項の数７(全 20 頁)

(21)出願番号　特願平4－235828

(22)出願日　平成４年(1992)９月３日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地

(71)出願人　391002409
日立システムエンジニアリング株式会社
東京都大田区大森北３丁目２番16号

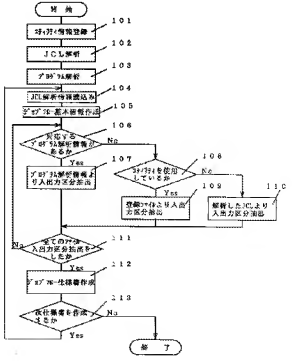
(72)発明者　秋庭　真一
神奈川県川崎市幸区鹿島田890番地の12
株式会社日立製作所ビジネスシステム開発センタ内

(74)代理人　弁理士　鈴木　誠

最終頁に続く

(54)【発明の名称】　ジョブフロー仕様書自動作成方法

(57)【要約】
【目的】ジョブ制御文やソースプログラム、データベース定義仕様等から、正確なジョブフロー仕様書を自動的に作成することを可能にする。
【構成】　プログラム解析が行えないユティリティに関するファイル入出力区分を予め登録する（１０１）。ジョブ制御文の解析（１０２）、ソースプログラムの解析（１０３）を行い、仕様書自動作成の対象となるジョブ制御文を入力し（１０４）、ジョブフロー基本情報を作成する（１０５）。そして、入力したジョブ制御文中のロードモジュール名より、対応するソースプログラム解析情報、ユティリティ情報またはジョブ制御文解析情報のいずれかよりファイル入出力区分を抽出し（１０６～１１１）、ジョブフロー仕様書を自動的に作成する（１１２）。



1

【特許請求の範囲】

【請求項1】　ジョブ制御文とソースプログラムを解析してジョブ制御文中の各ロードモジュールの実行順序と使用するファイルの入出力区分を判定して、ジョブ内のプログラムの実行順序とファイルの入出力を含む図式表現のジョブフロー仕様書を計算機の支援により自動作成する作成方法において、

予め少なくともユティリティ名とその使用ファイル名、ファイル入出力区分を示すユティリティ情報を記憶し、ジョブ制御文の解析によって得られるロードモジュール名よりユティリティであるかを前記情報より識別し、ユティリティと識別したロードモジュールに対しては、該情報より使用ファイルのファイル入出力区分を決定してジョブフロー仕様書を作成することを特徴とするジョブフロー仕様書自動作成方法。

【請求項2】　ジョブ制御文中のファイルの作成・削除・保持を示す情報の組合せ毎に、ファイル入出力区分を定めた情報を蓄積しておき、ロードモジュール名と同じ名称を持つソースプログラムがない場合は、該情報から当該ロードモジュールの使用ファイルの入出力区分を決定し、ジョブフロー仕様書を作成することを特徴とする請求項1記載のジョブフロー仕様書自動作成方法。

【請求項3】　データベース定義仕様を解析して、該ロードモジュールが入出力するデータベース名と入出力区分情報を得て、データベースに関する情報をジョブフロー仕様書に出力することを特徴とする請求項1または2記載のジョブフロー仕様書自動作成方法。

【請求項4】　ジョブ制御文中のロードモジュールに対するソースプログラム名を対話により指定し、ソースプログラムの解析情報から使用ファイルの入出力区分を取得することを特徴とする請求項1、2または3記載のジョブフロー仕様書自動作成方法。

【請求項5】　ジョブ制御文の解析で抽出したロードモジュール名がユティリティであるかを示す情報、対応するプログラム解析情報の有無を示す情報、データベースを使用の有無を示す情報の少なくともいづれかを予め作成してから、該情報を利用してジョブフロー仕様書を作成することを特徴とする請求項1、2、3または4記載のジョブフロー仕様書自動作成方法。

【請求項6】　請求項1、請求項2、請求項3、請求項4および請求項5記載のジョブフロー仕様書自動作成方法において、作成された複数のジョブフロー仕様書情報に基づき、ファイルとジョブまたはファイルとプログラムとの関連情報を作成して出力することを特徴とするジョブフロー仕様書自動作成方法。

【請求項7】　請求項1、請求項2、請求項3、請求項4、請求項5および請求項6記載のジョブフロー仕様書自動作成方法において、作成されたジョブフロー仕様書に対し、必要に応じて該仕様書を編集し、編集された該仕様書から新しいジョブ制御文を自動作成することを特徴とするジョブフロー仕様書自動作成方法。

2

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、ソフトウエア開発の計算機による支援システムに係り、特にソフトウエアの保守・理解、更に新規ソフトウエア開発作業に好適なジョブフロー仕様書の自動作成方法に関する。

【0002】

【従来の技術】一般にソフトウエアの開発では、開発すべきソフトウエアの具備すべき機能を定め、その機能を実現するソフトウエアを段階的に詳細化して設計を進め、最終的にプログラムを作成する方法が用いられている。

【0003】設計結果は、詳細化の各段階ごとに設計仕様書として記述され、次の設計段階では、前の段階で作成された仕様書の各要素を詳細化した仕様書が作成される場合が一般である。

【0004】こういった方法では、新規にソフトウエアを開発する場合には有効であるが、これを用いて開発したソフトウエアの保守を行うには、次のような問題点がある。すなわち、初期の段階で作成された上位仕様書と、これを詳細化して作成されたソフトウエア生産物（プログラム、ジョブ制御文、下位仕様書）と内容が一致しない場合が生じる。この場合、仕様書は実際のプログラムやジョブ制御文の内容を表さなくなり、保守が困難になる。

【0005】このような問題を解決するには、詳細化された仕様書、ソースプログラム、ジョブ制御文などの下位ソフトウエア生産物から上位仕様書を作成することにより、上位の仕様書が実際のソースプログラムやジョブ制御文の内容と一致させる必要がある。このための従来技術としては、例えば特願平1－237726号がある。ここでは、複数の詳細化された仕様書あるいはソースプログラムやジョブ制御文などのソフトウエア生産物を解析して、その中の上位仕様書に含むべき仕様情報を抽出し、この仕様情報を上位仕様書の形式に変換している。

【0006】

【発明が解決しようとする課題】上記特願平1－237726号は、下位のソフトウエア生産物から上位のソフトウエア生産物を作成する基本的な方法を示しているが、ジョブ内のソースプログラムの実行順序や、ファイルの入出力を表すジョブフロー仕様書を作成する場合、以下のような問題点が残されている。

【0007】(1)　ジョブ内で実行されるソースプログラムがユティリティの場合は、対応するソースプログラムを解析することができないので、正確なジョブフロー仕様書を自動作成することができない。

【0008】(2)　ソースプログラムが紛失している場合なども、ソースプログラムを解析して、使用するファ

イルの入出力区分を判定させることができないので、正確なジョブフロー仕様書を自動作成することができない。

【0009】(3) データベースを使用している場合、ジョブ制御文やソースプログラムの解析だけではそのデータベースの入出力が判定できないので、正確なジョブフロー仕様書を自動作成することができない。

【0010】(4) ジョブ制御文中で記述されているロードモジュール名とソースプログラム名が一致しない場合、対応するソースプログラムを見つけることができないので、正確なジョブフロー仕様書を自動作成することができなかい。

【0011】(5) 大量の既存ソフトウエア資源から上位の仕様書を作成する場合、処理効率が悪い。

【0012】(6) 作成するのは上位仕様書のみであり、保守や理解支援、変更影響範囲の洗い出し作業を効率的に行えるようにするための十分な情報を出力することができない。

【0013】(7) 作成されたジョブフロー仕様書を編集して、新規にジョブ制御文を自動作成することができなかい。

【0014】本発明の第1の目的は、正確なジョブフロー仕様書を自動的に作成することを可能にすることにある。

【0015】本発明の第2の目的は、処理効率良くジョブフロー仕様書の自動作成を行えるようにすることである。

【0016】本発明の第3の目的は、保守や理解支援、変更影響範囲の洗い出し作業を効率的に行えるようにするための情報を出力することにある。

【0017】本発明の第4の目的は、作成されたジョブフロー仕様書の編集や、該編集された該仕様書から新しいジョブ制御文を自動的に作成することを可能にすることにある。

【0018】

【課題を解決するための手段】請求項1の発明は、ジョブ制御文やプログラムから上位仕様情報であるジョブフロー仕様書を作成する際に、ユティリティに関して少なくともユティリティ名、その使用ファイル名、ファイル入出力区分情報を持つ情報を予め蓄積しておき、ジョブ制御文中に記述されているロードモジュール名が該情報のユティリティ名と一致した場合には、該情報中のファイル入出力区分情報を用いて、ジョブフロー仕様書を自動作成するものである。

【0019】請求項2の発明は、ジョブ制御文中のファイルの作成・削除・保持を示す情報の組合せに対し、各組合せにおいてのファイル入出力区分を定めたテーブルを予め蓄積しておき、ロードモジュール名と同じ名称を持つソースプログラムやユティリティが無い場合は、該テーブルから使用ファイルの入出力区分を決定し、ジョブフロー仕様書を自動作成するものである。

【0020】請求項3の発明は、予めデータベース定義仕様を解析しておき、ジョブ制御文またはソースプログラムとデータベース定義仕様の解析情報からデータベースを使用していると判断された時には、該解析情報より使用するデータベース名と入出力区分の情報を得て、ジョブフロー仕様書を自動作成するものである。

【0021】請求項4の発明は、ジョブ制御文の解析によって得られるロードモジュールに対応するプログラム名をディスプレイ画面上で対話形式で指定し、指定されたソースプログラムの解析情報から使用ファイルと入出力区分を取得し、ジョブフロー仕様書を自動作成するものである。

【0022】請求項5の発明は、ジョブ制御文、ソースプログラム、データベースの解析情報とユティリティに関する情報より、ジョブ制御文解析で抽出したロードモジュール名がユティリティであるを示す情報、対応するプログラム解析情報の有無やデータベースを使用の有無を示す情報を自動的に作成してから、該情報を利用してジョブフロー仕様書を自動作成するものである。

【0023】請求項6の発明は、上記のようにして作成された複数のジョブフロー仕様書情報に基づき、ファイルとそれを使用しているジョブの関連情報、またはファイルとそのファイルを使用しているプログラムの関連情報を作成して出力するものである。

【0024】請求項7の発明は、上記のようにして作成されたジョブフロー仕様書をディスプレイ画面上で対話形式で編集し、その編集されたジョブフロー仕様書に基づき新しいジョブ制御文を自動的に作成するものである。

【0025】

【作用】第1の目的に対しては、ジョブ制御文やプログラムから上位仕様情報であるジョブフロー仕様書を作成する際に、ユティリティに関して少なくともユティリティ名、その使用ファイル名、ファイル入出力区分情報を持つ情報を蓄積しておき、ジョブ制御文中に記述されているロードモジュール名が該情報のユティリティ名と一致した場合には、該情報中のファイル入出力区分を決定させることで、ユティリティを使用しているの場合においても、正確なジョブフロー仕様書が自動作成できる。

【0026】また、ジョブ制御文中のファイルの作成・削除・保持を示す情報の組合せに対し、各組合せにおいてのファイル入出力区分を定めた情報を蓄積しておき、ジョブ制御文中に記述されているロードモジュール名と同じ名称を持つソースプログラムが無いロードモジュールに対しては、該情報から使用ファイルの入出力区分を決定させることで、ロードモジュールに対する解析情報が無い場合においても、正確なジョブフロー仕様書が自動作成できる。

【0027】また、予めデータベース定義仕様を解析し

ておき、該解析情報より使用するデータベース名と入出力区分を得ることで、データベースを使用している場合においても、正確なジョブフロー仕様書が自動作成できる。

【００２８】また、ロードモジュール名に対応するプログラム名をディスプレイ画面上で対話形式で指定することができるため、ロードモジュール名と名称が違っているためにソースプログラムと対応付けができなかった場合においても、指定されたソースプログラム解析情報から使用するファイルの入出力区分を決定し、自動的にジョブフロー仕様書を作成することができる。

【００２９】第２の目的に対しては、ジョブ制御文、ソースプログラム、データベースの解析情報とユティリティに関する情報より、ジョブ制御文の解析で抽出したロードモジュール名がユティリティであるかを示す情報、対応するプログラム解析情報の有無を示す情報やデータベースの使用の有無を示す情報を予め作成し、該テーブルを利用してジョブフロー仕様書を作成することで、それぞれの情報に対して行う検索処理を必要最小限にとり、ジョブフロー仕様書が効率良く自動作成される。

【００３０】第３の目的に対しては、ジョブ制御文、ソースプログラム、データベースの解析情報とユティリティに関する情報より、ジョブ制御文解析で抽出したロードモジュール名がユティリティであるか、対応するプログラム解析情報の有無やデータベースを使用の有無を示す情報からファイルとそれを使用しているジョブの関連情報及びファイルとそのファイルを使用しているプログラムの関連情報を作成し、出力することで、あるソースプログラムまたはジョブ制御文に修正が行われた場合、他のどこに影響があるかを分析するため情報が出力され、保守や変更影響範囲の洗い出し作業を効率的に行うことができるようになる。

【００３１】第４の目的に対しては、下位ソフトウエア生産物であるジョブ制御文、ソースプログラム、データベース定義仕様等より作成したジョブフロー仕様書をディスプレイ画面上で編集し、編集された該仕様書からジョブ制御文を自動作成させることを可能にするため、システムの再開発／再構築作業の効率化に役立つことができるようになる。

【００３２】

【実施例】図２は、本発明に係わるジョブフロー仕様書自動作成方法を達成するシステムの構成例の概略ブロック図を示す。図２において、１はプログラムに基づく逐次処理機能を有するＣＰＵであり、ここでは、後述の図１、図１１、図１７あるいは図２３の手順の実行を制御する。２はキーボードとマウスとディスプレイ画面を有する対話端末であり、該システムを用いる作業者が指示を入力したり、結果を該作業者に表示するために用いる。３は図１、図１１、図１７あるいは図２３の手順を実現するためのプログラム、処理の中間結果などが格納される主メモリである。４は該システムで用いられる既存ソフトウエア資源が格納されているファイルである。５は作業者によって登録された情報が格納されているファイルである。６は該システムによって作成されたジョブフロー仕様書等の情報が格納されるファイルである。７はプリンタ装置であり、ここでは該システムによって作成されたジョブフロー仕様書等が出力される。以下、本発明の各実施例について詳述する。

【００３３】＜実施例１＞図１は、本発明に係わるジョブフロー仕様書自動作成方法の第１の実施例の処理手順を示すフローチャートである。以下に、図１のフローチャートに基づき第１の実施例を説明する。

【００３４】まず、解析対象のジョブ制御文（以下、ＪＣＬと称す）で使用しているユティリティに関する情報を予め登録する（ステップ１０１）。その登録情報の例を図３に示す。３０１はユティリティの名称で、そのユティリティが使用するＤＤ名（３０２）と各々のＤＤのファイル入出力区分（３０３）を設定する。また、そのファイルが一時的であるときは、３０４に示すようなフラグを付ける。

【００３５】次に、該ジョブフロー仕様書自動作成の対象となる全てのＪＣＬの解析（ステップ１０２）及びソースプログラムの解析（ステップ１０３）を行う。

【００３６】図４は解析対象のＪＣＬの例であり、４０１はジョブ名、４０２はジョブステップ名、４０３はロードモジュール名、４０４は物理ファイル名、４０５はＤＤ名、４０６はＤＩＳＰオペランドのパラメータを示す。図４のＪＣＬの解析結果は、図５に示したようなテーブルに格納される。例えばジョブ名（５０１）には４０１の「ＪＯＢ１」、ジョブステップ名（５０２）には４０２の「ＳＴＥＰ１」、ロードモジュール名（５０３）には４０３の「ＰＲＯＧ０１」、物理ファイル名（５０４）には４０４の「ＳＡＭＰＬＥ１」、ＤＤ名（５０５）には４０５の「ＤＤ１」をそれぞれ設定し、さらに、ファイルの入出力区分（５０６）には、図６のテーブルに示すＤＩＳＰオペランドのパラメータの組合せによって定義された情報に従って設定する。図６で、６０１はＤＩＳＰオペランドのパラメータの組合せによって設定するファイルの入出力区分を示し、６０２はそのファイルが一時的なものであるかを示すフラグである。該テーブルの情報も予め登録／編集が可能である。

【００３７】図７は解析対象のプログラムの例であり、７０１はプログラム名、７０２は論理ファイル名、７０３はＤＤ名、７０４はレコード名、７０５はファイルの入出力区分を示す。図７の解析結果は、図８に示したようなテーブルに格納される。例えば、プログラム名（８０１）には７０１の「ＰＲＯＧ０１」、ファイル名（８０２）には７０２の「ＩＮＦＩＬＥ」、ＤＤ名（８０３）には７０３の「ＤＤ１」、レコード名（８０４）には７０４からの「ＩＮ－ＲＥＣ」、ファイルの入出力区

分（８０５）には、図７の７０５に該レコードのＲＥＡＤが示されているので入力と見なし「Ｉ」を設定する。なお、ＷＲＩＴＥの場合は出力と見なし、ファイルの入出力区分には「Ｏ」が設定される。

【００３８】以上の解析が終了後、ジョブフロー自動作成を行うＪＣＬ解析情報（図５）を読み込む（ステップ１０４）。次に、ジョブフロー仕様書作成のためのジョブフロー基本情報を作成する（ステップ１０５）。ジョブフロー基本情報の例を図９に示す。図９は図５に対応するもので、例えば、ジョブ名（９０１）には５０１の「ＪＯＢ１」、ジョブステップ名（９０２）には５０２の「ＳＴＥＰ１」を設定し、ロードモジュール名（９０３）には５０３の「ＰＲＯＧ０１」、物理ファイル名（９０４）には５０４の「ＳＡＭＰＬＥ１」、ＤＤ名（９０５）には５０５の「ＤＤ１」を設定する。

【００３９】次に、図９のファイル入出力区分（９０６）を設定するために、ロードモジュール名（９０３）と一致するプログラム名を持つプログラム解析情報（図８）があるかチェックする（ステップ１０６）。もし一致するプログラム名（８０１）が存在する場合は、そのプログラム解析情報で示すファイルの入出力区分（８０５）を抽出し、図９のファイル入出力区分（９０６）に追加する（ステップ１０７）。もしロードモジュール名（９０３）と一致するプログラム名を持つプログラム解析情報が無かった場合は、そのロードモジュール名９０３と一致するユティリティ名がステップ１０１で設定されたテーブル（図３）に登録されているかをチェックする（ステップ１０８）。一致するユティリティ名（３０１）が存在する場合は、テーブル中の３０３で示すファイルの入出力区分を抽出し、図９のファイル入出力区分（９０６）に追加する（ステップ１０９）。この時、テーブル中の一時ファイル区分に一時ファイル判定フラグ「Ｙ」（３０４）が指定されていた場合は、図９の一時ファイル区分（９０７）に「Ｙ」を設定する。また、ステップ１０８で一致するユティリティ名が存在しなかった場合は、ＪＣＬ解析情報（図５）のファイル入出力区分（５０６）から抽出し、図９のファイル入出力区分（９０６）に追加する（ステップ１１０）。

【００４０】ステップ１０６～１１０の処理について、入力したＪＣＬ解析情報に関して全てのファイル入出力区分を抽出するまで繰り返す（ステップ１１１）。そして、全てのファイル入出力区分を抽出すると、該ジョブフロー基本情報よりジョブフロー仕様書を作成する（ステップ１１２）。その後、次のジョブフロー仕様書を作成する場合はステップ１０４に戻り（ステップ１１３）、該ジョブフロー自動作成の対象となる全てのＪＣＬ解析情報に対する処理を繰り返し実行し、ジョブフロー仕様書を作成する。

【００４１】図１０に、作成したジョブフロー仕様書の例を示す。ここで、１００１は一般ファイルを示し、１００２は一般プログラムを示す。また、１００３はユティリティを示し、１００４は一時ファイル、１００５は出力装置であるプリンタを示す。また、１００６は図９の９０４の物理ファイル名「ＳＡＭＰＬＥ１」であり、１００７は９０５の該ファイルを入力するときのＤＤ名「ＤＤ１」を示し、１００８は９０２のジョブステップ名「ＳＴＥＰ１」、１００９は９０３のロードモジュール名「ＰＲＯＧ０１」を示す。

【００４２】本実施例によれば、従来ＪＣＬだけでプログラム解析情報が得られないユティリティなどの不明確であったファイルの入出力区分を、予め一度定義しておくだけで、ファイルの入出力区分を明確にしたジョブフロー仕様書が随時自動作成されるようになる。このため、作成した仕様情報を基に現システムの保守・理解に役立つ情報を提供できるようになる。

【００４３】＜実施例２＞図１１は本発明に係るジョブフロー仕様書自動作成方法の第２の実施例の処理手順を示すフローチャートである。本実施例では、第１の実施例に加え、使用するファイルがデータベースか、一般ファイルかを区別して、自動作成されたジョブフロー仕様書においてデータベースの使用状況を視覚的に表現させることを可能にするものである。図１１において、図１のステップ１０５が１１０４に、ステップ１１２が１１０５に変わり、１１０１、１１０２、１１０３のステップが追加された以外は図１と同じである。

【００４４】まず、第１の実施例と同様に、ユティリティ情報登録、ＪＣＬの解析ソースプログラムの解析まで行い、次にデータベース定義仕様の解析を行う（ステップ１１０１）。図１２は、あるデータベース定義仕様の例であり、１２０１はアクセス定義仕様書名、１２０２はデータベース名、１２０３はアクセス権限情報、１２０４はプログラム名、１２０５は物理ファイル名を示す。この解析結果は、図１３に示すようなテーブルに格納される。例えばアクセス仕様名称（１３０１）には１２０１の「ＲＳＶＲＤ」、データベース名（１３０２）には１２０２の「ＳＡＭＰＬＥＤＢ」、物理ファイル名（１３０３）には１２０５の「ＥＸＡＭ１」、使用プログラム名（１３０４）には１２０４の「ＰＲＯＧ０２」が設定される。そして、入出力区分（１３０５）には、１２０３の「ＳＨＲＵＰＤ」により、読み込みも書き込みも可能と定義されているので、入力と出力「ＩＯ」と設定される。

【００４５】次に、ジョブフロー自動作成を行うＪＣＬ解析情報を読み込み（ステップ１０４）、ジョブフロー基本情報の作成を行う（ステップ１１０４）。ジョブフロー基本情報は図９とほぼ同様であるが、図９に示される物理ファイル名（９０４）とＤＤ名（９０５）は設定しない。そして、ステップ１１０２で、データベースを使用しているか否かを、ＪＣＬ中のロードモジュール名がデータベースを使用する場合に用いられる特定名称で

あるか、またデータベース定義仕様で定義されているソースプログラム名とソースプログラム解析で得られるプログラム名が一致しているかで判断し、使用している場合には、ステップ１１０３で、ジョブフロー基本情報に図１３の１３０２のデータベース名と１３０３の物理ファイル名と１３０４の入出力区分を設定する。一致していない場合は、実施例１と同様な処理手順で、ファイル入出力区分を抽出する。

【００４６】図１４は、ＪＣＬ中でデータベースを使用する場合に用いられる特定名称がロードモジュールに記述されている場合の例を示しており、該データベース解析情報を取り込んで作成されたジョブフロー基本情報を図１５に示す。図１４の１４０１で囲まれているのが、該データベースを使用するＪＣＬ部である。該データベースを使用するＪＣＬ部について、図１５のジョブフロー基本情報では、ジョブ名（１５０１）には１４０２の「ＪＯＢ２」、ジョブステップ名（１５０２）には１４０３の「ＳＴＥＰ２」、ロードモジュール名（１５０３）には１４０４の「ＰＤＭＢＴＣＨ」、そして、データベース名（１５０４）には、図１３のデータベース解析情報の１３０２の「ＳＡＭＰＬＥＤＢ」、物理ファイル名（１５０５）には１３０３の「ＥＸＡＭ１」、入出力区分（１５０６）には１３０５の「ＩＯ」が設定される。

【００４７】該ジョブフロー基本情報よりジョブフロー仕様書を自動作成する（ステップ１１０５）。このステップ１１０５では、図１のジョブフロー仕様書作成（１１２）にデータベースを使用している場合に親資的に明示させることを追加している。図１６に該ジョブフロー仕様書を例示する。１６０１はデータベースを一般ファイルと区別をして表現したもので、１６０２は１５０４のデータベース名を示し、１６０３は１５０５のデータベースで使用している物理ファイル名を示し、１６０４は１５０６の入出力区分を示している。

【００４８】本実施例によれば、図１０に示した第１の実施例のジョブフロー仕様書に加え、データベースの使用情報を含むジョブフロー仕様書を自動作成することができる。

【００４９】＜実施例３＞図１７は、本発明に係るジョブフロー仕様書自動作成方法の第３の実施例の処理手順を示すフローチャートである。図１１で示した第２の実施例では、ファイル入出力区分を抽出するためにプログラム解析情報があるか、登録したユティリティ情報があるか、またデータベースを使用しているのかといった各情報毎に検索処理を行っている。そのため、大量に既存ソフトウエア資源が存在する場合において検索に要する無駄な時間がかなり生じると考えられる。そこで本実施例では、第２の実施例よりも処理効率を向上させ、更により正確なジョブフロー仕様書を自動作成させるために、全解析が終了した後、各解析情報の関連のを持つテーブルを自動的に作成する。ジョブフロー仕様書作成時には、該テーブルより関係付いた情報が格納されているファイルのみ検索してファイル入出力区分を抽出し、ジョブフローを自動的に作成できるように改良したものである。このため、図１７では、図１１に１７０１、１７０２、１７０３、１７０４、１７０６、１７０７が追加され、１１０２、１０６、１０８のステップを削除し、１０７、１０９、１１０３のステップを１７０５で示す１つのステップにまとめてある。

【００５０】まず、第２の実施例と同様に、データベース解析まで行い（ステップ１０１，１０２，１０３，１１００）、解析終了後に各情報の関連を示す情報テーブルを自動的に作成する（ステップ１７０１）。図１８に該テーブルを例示する。図１８において、１８０１にはＪＣＬ解析によって得られるロードモジュール名（図５の５０３）、１８０２にはそれに対応するプログラム名（図８の８０１）もしくはユティリティ名（図３の３０１）が設定される。ロードモジュールとプログラムの対応は、プログラムが格納されているメンバ名もしくはプログラム中に記述されているプログラムＩＤによって対応付けが行われる。該テーブルを作成した後に、関連情報を編集する必要があるか判定し（ステップ１７０２）、編集の必要があれば、ディスプレイ画面上で対話形式でプログラム名（１８０２）とプログラム区分（１８０３）、使用しているデータベース名（１８０４）を編集する（ステップ１７０３）。

【００５１】次に、ジョブフロー基本情報を作成し（ステップ１１０４）、抽出したロードモジュール名に関連する情報があるかを上記テーブルから判定し（ステップ１７０４）、対応する情報がある場合は、該テーブル内にある情報から該ロードモジュールに対応する情報が格納されているファイルのみを検索し、ファイル名とそのファイル入出力区分を抽出する（ステップ１７０５）。関連する情報が無い場合は、直接ＤＩＳＰオペランドの組合せによって定義されたテーブルからファイル名とそのファイル入出力区分を抽出する（ステップ１１０）。これを１ロードモジュール内のすべてのファイルに対してファイル名とそのファイル入出力区分を抽出する（ステップ１７０６）。

【００５２】次に、各情報の関連（ＪＣＬ／ソース、ＪＣＬ／ＤＢ）を示す情報テーブルを使用した場合と使用しない場合の処理の違いについて説明する。図１９は、第２の実施例で示された手順により、ジョブフロー自動作成する例を示し、図２０は、本実施例により作成された該テーブルを使用してジョブフロー仕様書を自動作成する手順を示している。本例では、ロードモジュール名「ＴＥＳＴＰＲＯＧ」に対応する解析情報が全くない場合の例を示す。

【００５３】図１９では、ジョブフロー自動作成の対象となるＪＣＬ解析情報からロードモジュール名を抽出し

（１９０１）、次にデータベースを使用しているかデータベース解析情報を検索し（１９０２）、無いと判断されると、次に該ロードモジュール名と一致するプログラム名を持つプログラム解析情報が存在するか検索し（１９０３）、無いと判断されると、次にユティリティとして登録されているか検索を行う（１９０４）。本例のように、該ロードモジュールに対応する情報が無かった場合は、最後にＪＣＬ解析中にあるＤＩＳＰオペランドに定義されたファイル入出力区分を抽出し（１９０５）、ジョブフロー仕様書を自動作成する（１９０６）。

【００５４】図２０では、まず、ジョブフロー自動作成の対象となるＪＣＬ解析情報からロードモジュール名を抽出（２００１）し、本実施例で作成されたテーブル２０００から関係する情報があるかを判定する（２００２）。本例では、該ロードモジュールに対応する情報が存在しないので、直接ＤＩＳＰオペランドに定義されたファイル入出力区分を抽出し（２００３）、ジョブフロー仕様書を自動作成する（２００４）。以上のような手順の違いにより処理効率の向上が図れる。

【００５５】また、図１７のステップ１７０７において、先に作成された複数の読テーブル情報より図２１のような物理ファイルとジョブの関連情報や、図２２に示すような論理ファイルとプログラムの関連情報を作成して出力する。図２１において、物理ファイル名（２１０１）にはＪＣＬ解析で得られる図５の５０４の「ＳＡＭＰＬＥ１」、同様にジョブ名（２１０２）には５０１の「ＪＯＢ１」、ジョブステップ名（２１０３）には５０２の「ＳＴＥＰ１」、ＤＤ名（２１０４）には５０５の「ＤＤ１」、ファイルの入出力区分（２１０５）には５０６の「Ｉ」が設定され、複数の該情報がプリンタ装置に出力される。該出力情報によって、どのジョブのどのジョブステップでどのファイルがアクセスされるか明確にできる。また、図２２のように、論理ファイル名（２２０１）にはプログラム解析で得られる図８の８０２の「ＩＮＦＩＬＥ」、同様にプログラム名（２２０２）には８０１の「ＰＲＯＧ０１」、ＤＤ名（２２０４）には８０３の「ＤＤ１」、ファイルの入出力区分（２２０５）には８０５の「Ｉ」、プログラム区分（２２０３）については図１８の１８０３の「ＳＯＵＲＣＥ」が設定され、複数の該情報がプリンタ装置に出力される。該出力情報によって、どのプログラムどのファイルがアクセスされるか明確にできる。これらの情報により、使用しているファイルに関する修正が行われる時などに影響する範囲を早期に発見することが可能となる。

【００５６】本実施例によれば、第２の実施例よりも処理を効率良く行うことができる。また、ＪＣＬ内のロードモジュール名とソースプログラム名が一致しない情報に関しても、ロードモジュール名と対応するプログラム名をディスプレイ画面上で対話形式で指定することができるため、指定したプログラムのファイル入出力区分を自動的に取り入れた、より正確なジョブフロー仕様書を自動作成することができる。

【００５７】＜実施例４＞図２３は、本発明に係るジョブフロー仕様書自動作成方法の第４の実施例の処理手順を示すフローチャートである。本実施例は、図１７に２３０１、２３０２、２３０３のステップを加えたものである。本実施例では、第１、第２、第３の実施例で自動作成されたジョブフロー仕様書に対して編集を行い、編集されたジョブフロー仕様書より新しいＪＣＬを自動的に作成できるように改良したものである。

【００５８】図２３において、自動作成されたジョブフロー仕様書を基に該仕様書を編集するか判定し（ステップ２３０１）、編集する必要があると判断された場合は、ディスプレイ画面上でマウスやキーボードを用いて編集を行う（ステップ２３０２）。自動作成された編集前のジョブフロー仕様書を図１０とし、該仕様書に編集を加えたジョブフロー仕様書の例を図２４に示す。この編集では、図１０で示されるジョブ名「ＪＯＢ１」（９０１）のＪＣＬに対して、ジョブステップ「ＳＴＥＰ１」（９０２）のロードモジュール名「ＰＲＯＧ０１」（９０３）に物理ファイル名「ＴＲＡＮ１」（２４０２）、ＤＤ名「ＤＤ５」（２４０３）のファイルを入力する情報を自動作成されたジョブフロー仕様書に追加している。

【００５９】次に、編集されたジョブフロー仕様書からＪＣＬを自動的に作成する（ステップ２３０３）。自動的に作成されたＪＣＬの例を図２５に例示する。２５０１の「ＪＯＢ１」と２５０２の「ＳＴＥＰ１」と２５０３の「ＰＲＯＧ０１」は、自動作成したジョブフロー仕様書が既に持っていたＪＣＬ解析情報（図５）であり、本例で編集されて自動的に作成されたＪＣＬは、２５０５のＤＤ名に図２４の２４０３の「ＤＤ５」、２５０４の物理ファイル名に２４０２の「ＴＲＡＮ１」、ファイル入出力区分は入力であるため、「ＤＩＳＰ＝ＳＨＲ」（２５０６）が図１０のジョブフロー仕様書を作成したＪＣＬに追加される。

【００６０】本実施例によれば、ＪＣＬやプログラム等から自動作成したジョブフロー仕様書を編集し、編集した該仕様書から新たにＪＣＬを自動作成させることができる。

【００６１】

【発明の効果】

(1)　請求項１の発明によれば、ジョブ内のユティリティの使用ファイルの入出力が判定できるので、正確なジョブフロー仕様書が自動作成される。

【００６２】(2)　請求項２の発明によれば、対応するソースプログラムのないロードモジュールに対して、使用ファイルの入出力が判定できるので、正確なジョブフロー仕様書が自動作成される。

【００６３】(3)　請求項３の発明によれば、使用する

データベースに対して、入出力が判断できるので、正確なジョブフロー仕様書が自動作成される。

【0064】(4) 請求項4の発明によれば、ロードモジュールに対し、対応するソースプログラム名を指定し、指定されたソースプログラムの解析情報から使用するファイルの入出力区分が判定できるため、正確なジョブフロー仕様書が自動作成される。

【0065】(5) 請求項5の発明によれば、予めロードモジュールとソースプログラム、データベース、ユティリティの関連情報テーブルを作成してから、ジョブフロー仕様書を作成するので、ジョブフロー仕様書自動作成の処理効率を上げることができる。

【0066】(6) 請求項6の発明によれば、ファイル／ジョブ関連情報、ファイル／プログラム関連情報を作成し、出力するので保守や理解支援、変更影響範囲の洗い出し作業を効率的に行うことができる。

【0067】(7) 請求項7の発明によれば、作成されたジョブフロー仕様書の修正や修正されたジョブフロー仕様書がら新しいＪＣＬを自動作成できるので、システムの再開発／再構築の効率化に役立つことができる。

【図面の簡単な説明】

【図1】本発明の第1の実施例の処理手順を示すフローチャートである。

【図2】本発明のジョブフロー仕様書自動作成方法を達成するシステムの構成例を示すブロック図である。

【図3】第1の実施例で使用するユティリティ登録情報の例である。

【図4】第1の実施例で使用するＪＣＬの例である。

【図5】第1の実施例で使用するＪＣＬ解析情報の例である。

【図6】第1の実施例で使用するＤＩＳＰオペランドの組合せ毎によるファイル入出力区分を決定させる定義情報の例である。

【図7】第1の実施例で使用するプログラムの例である。

【図8】第1の実施例で使用するプログラム解析情報の例である。

【図9】第1の実施例で作成されるジョブフロー基本情報の例である。

【図10】第1の実施例で作成されたジョブフロー仕様書の例である。

【図11】本発明の第2の実施例の処理手順を示すフローチャートである。

【図12】第2の実施例で使用するデータベース仕様定義の例である。

【図13】第2の実施例で使用するデータベース仕様定義の解析情報の例である。

【図14】第2の実施例で使用するＪＣＬの例である。

【図15】第2の実施例で作成されるジョブフロー基本情報である。

【図16】第2の実施例で作成されるジョブフロー仕様書の例である。

【図17】本発明の第3の実施例の処理手順を示すフローチャートである。

【図18】第3の実施例で作成する関連情報テーブルの例である。

【図19】第3の実施例における使用ファイルとその入出力区分抽出手順の例を示す図である。

【図20】第3の実施例における使用ファイルとその入出力区分抽出手順の例を示す図である。

【図21】ファイル／ジョブ関連情報の例である。

【図22】ファイル／プログラム関連情報の例である。

【図23】本発明の第4の実施例の処理手順を示すフローチャートである。

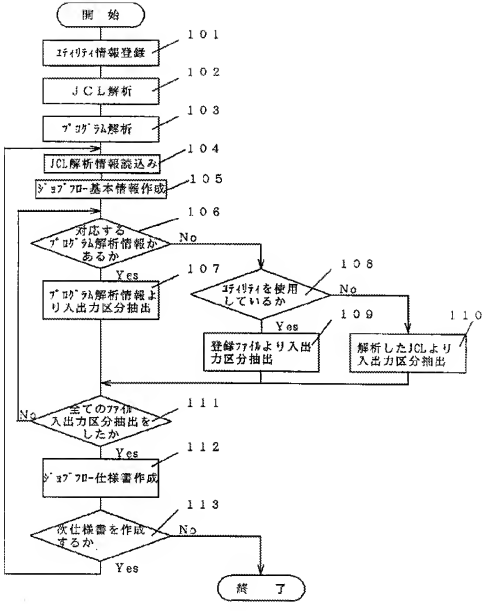
【図24】図10のジョブフロー仕様書の編集後のジョブフロー仕様書の例である。

【図25】図24のジョブフロー仕様書より自動作成されたＪＣＬの例である。

【符号の説明】

1　ＣＰＵ
2　対話端末
3　主メモリ
4　既存ソフトウエア資源格納ファイル
5　ユーザ登録情報ファイル
6　ジョブフロー基本情報ファイル
7　プリンタ装置
101　ユティリティ情報登録ステップ
102　ＪＣＬの解析ステップ
103　ソースプログラムの解析ステップ
104　ＪＣＬ解析情報入力ステップ
105,1104　ジョブフロー基本情報作成ステップ
107,109,110,1103,1705　ファイル入出力区分抽出ステップ
112,1105　ジョブフロー仕様書作成ステップ
1703　JCL／ソースプログラム、JCL／データベース関連情報作成ステップ
1707　ファイル／ジョブ、ファイル／プログラム情報作成ステップ
2302　ジョブフロー仕様書の編集ステップ
2303　ＪＣＬ作成ステップ

【図1】

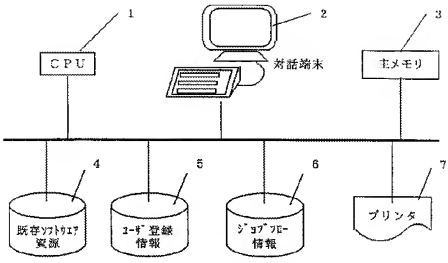


【図3】

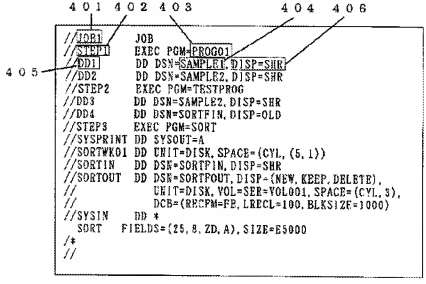
ユティリティ名	DD名	I/O区分	一時ファイル区分
SORT	SYSPRINT	0	Y
	SORTWK01	IO	
	SORTIN	I	
	SORTOUT	O	

301　302　303　304

【図2】



【図4】



【図8】

ﾌﾟﾛｸﾞﾗﾑ名	論理ﾌｧｲﾙ名	DD名	ﾚｺｰﾄﾞ名	I／O区分
PROG01	INFILE	DD1	IN-REC	I
PROG01	OUTFILE	DD2	OUT-REC	O

801　802　803　804　805

【図5】

501　502　503　504　505　506

ｼﾞｮﾌﾞ名	ｼﾞｮﾌﾞｽﾃｯﾌﾟ名	ﾛｰﾄﾞﾓｼﾞｭｰﾙ名	物理ﾌｧｲﾙ名	DD名	I/O区分	一時ﾌｧｲﾙ区分
JOB1	STEP1	PROGD1	SAMPLE1	DD1	I	
JOB1	STEP1	PROGD1	SAMPLE2	DD2	I	
JOB1	STEP2	TESTPROG	SAMPLE2	DD3	I	
JOB1	STEP2	TESTPROG	SORTFIN	DD4	O	
JOB1	STEP3	SORT		SYSPRINT	O	
JOB1	STEP3	SORT		SORTWK01	IO	
JOB1	STEP3	SORT	SORTFIN	SORTIN	I	
JOB1	STEP3	SORT	SORTFOUT	SORTOUT	O	

【図6】

第一ﾊﾟﾗﾒﾀ ＼ 第二ﾊﾟﾗﾒﾀ	DELETE	KEEP	PASS	CATLG	UNCATLG	なし
NEW	IO Y	O	O	O	O	DELETEと同じ
MOD	IO	O	O	O	O	KEEPと同じ
RNW	IO Y	O	O	O	O	DELETEと同じ
OLD	IO	O	O	O	O	KEEPと同じ
SHR	I	I	I	I	I	KEEPと同じ
なし	NEWと同じ					

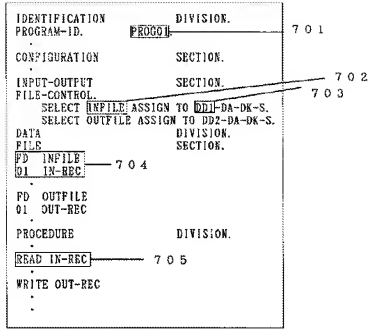
601　602

【図12】

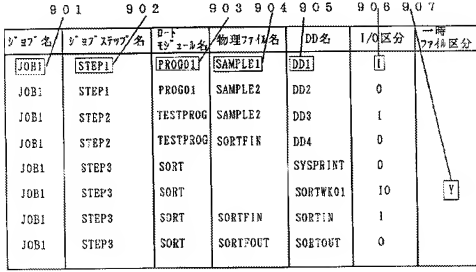
1201　1202　1203

```
ASB=RSVRD, DBM=SAMPLEDB, ACCESS=SHRUPD
  PROG=PROG02          1204
    .
    .
  DSN=EXAM1            1205
    LIST=EXAM1
    .
    .
  DSN=EXAM2
    LIST=EXAM2
  END
```

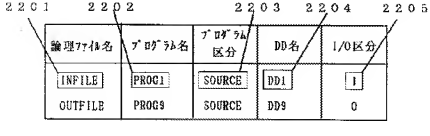
【図7】


IDENTIFICATION DIVISION.
PROGRAM-ID. PROG01.
701
CONFIGURATION SECTION.
INPUT-OUTPUT SECTION.
702
FILE-CONTROL.
703
SELECT INFILE ASSIGN TO DD1-DA-DK-S.
SELECT OUTFILE ASSIGN TO DD2-DA-DK-S.
DATA DIVISION.
FILE SECTION.
FD INFILE
01 IN-REC
704
FD OUTFILE
01 OUT-REC
PROCEDURE DIVISION.
READ IN-REC
705
WRITE OUT-REC

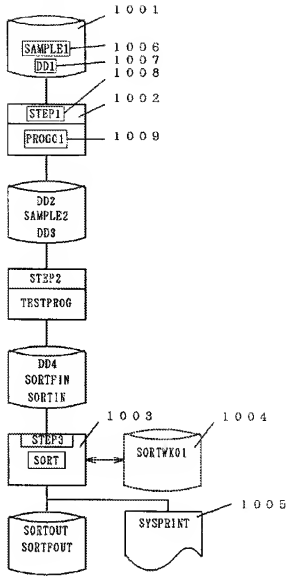
【図9】


901 902 903 904 905 906 907
ジョブ名
ジョブステップ名
ロードモジュール名
物理ファイル名
DD名
I/O区分
一時ファイル区分
JOB1 STEP1 PROG01 SAMPLE1 DD1 I
JOB1 STEP1 PROG01 SAMPLE2 DD2 O
JOB1 STEP2 TESTPROG SAMPLE2 DD3 I
JOB1 STEP2 TESTPROG SORTFIN DD4 O
JOB1 STEP3 SORT SYSPRINT O
JOB1 STEP3 SORT SORTWK01 IO Y
JOB1 STEP3 SORT SORTFIN SORTIN I
JOB1 STEP3 SORT SORTFOUT SORTOUT O

【図22】


2201 2202 2203 2204 2205
論理ファイル名
プログラム名
プログラム区分
DD名
I/O区分
INFILE PROG1 SOURCE DD1 I
OUTFILE PROG9 SOURCE DD9 O

【図１０】

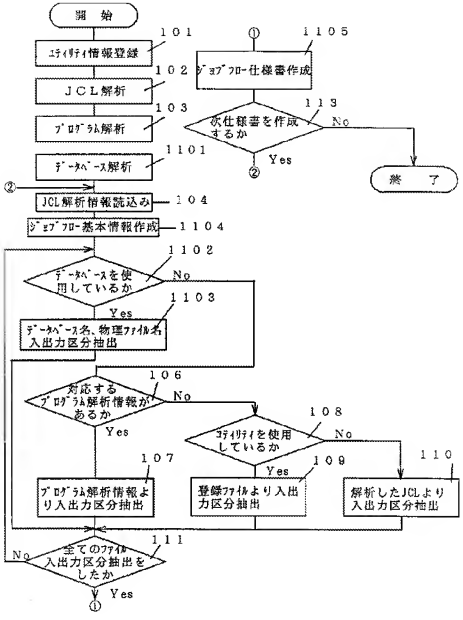


【図１３】

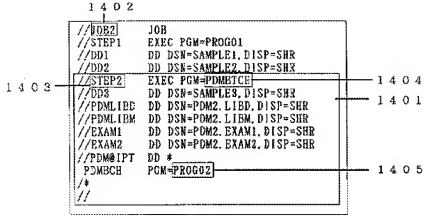
ｱｸｾｽ定義仕様書名	ﾃﾞｰﾀﾍﾞｰｽ名	物理ﾌｧｲﾙ名	使用ﾌﾟﾛｸﾞﾗﾑ名	I/O区分
RSVRD	SAMPLEDB	EXAM1	PROGO2	IO
RSVRD	SAMPLEDB	EXAM2	PROGO2	IO

１３０１　１３０２　１３０３　１３０４　１３０５

【図１１】


開　始
１０１
ユティリティ情報登録
１０２
ＪＣＬ解析
１０３
プログラム解析
１１０１
データベース解析
②
ＪＣＬ解析情報読込み
１０４
ジョブフロー基本情報作成
１１０４
１１０２
データベースを使用しているか
No
Yes
１１０３
データベース名、物理ファイル名入出力区分抽出
１０６
対応するプログラム解析情報があるか
No
Yes
１０７
プログラム解析情報より入出力区分抽出
１０８
ユティリティを使用しているか
No
Yes
１０９
登録ファイルより入出力区分抽出
１１０
解析したＪＣＬより入出力区分抽出
１１１
全てのファイル入出力区分抽出をしたか
No
Yes
①
①
１１０５
ジョブフロー仕様書作成
１１３
次仕様書を作成するか
No
Yes
②
終　了

【図14】

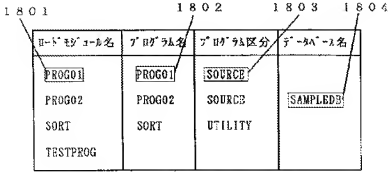


【図15】

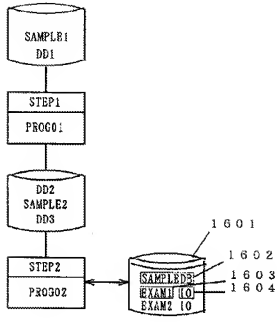
ｼﾞｮﾌﾞ名	ｼﾞｮﾌﾞｽﾃｯﾌﾟ名	ﾛｰﾄﾞﾓｼﾞｭｰﾙ名	ﾃﾞｰﾀﾍﾞｰｽ名	物理ﾌｧｲﾙ名	DD名	I/O区分	一時ﾌｧｲﾙ区分
JOB2	STEP1	PROGO1		SAMPLE1	DD1	I	
JOB2	STEP1	PROGO1		SAMPLE2	DD2	O	
JOB2	STEP2	PDMBTCH		SAMPLE2	DD3	I	
JOB2	STEP2	PDMBTCH	SAMPLEDB	EXAM1		IO	
JOB2	STEP2	PDMBTCH	SAMPLEDB	EXAM2		IO	

1501　1502　1503　1504　1505　1506

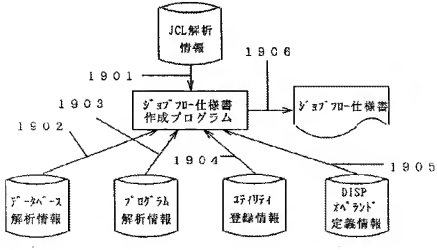
【図18】



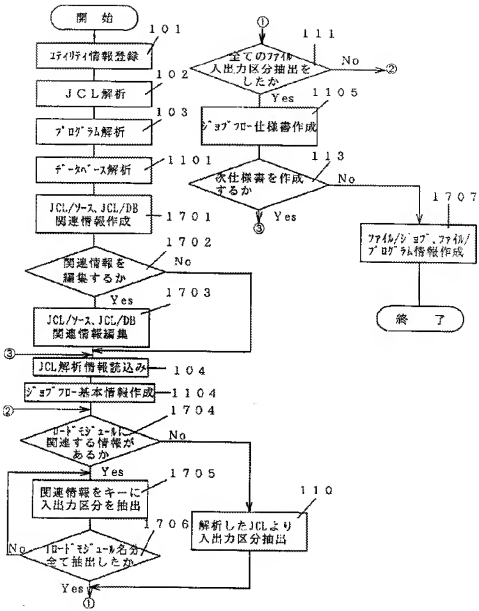
【図16】


SAMPLE1
DD1
STEP1
PROG01
DD2
SAMPLE2
DD3
STEP2
PROG02
1601
1602
SAMPLE3
EXAM1 IO
EXAM2 IO
1603
1604

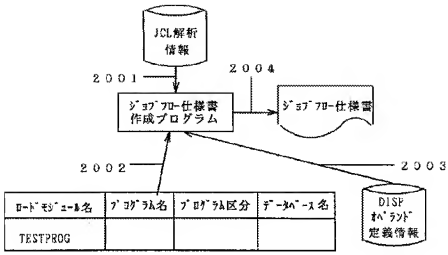
【図19】


JCL解析
情報
1906
1901
1903
ｼﾞｮﾌﾞﾌﾛｰ仕様書
作成プログラム
ｼﾞｮﾌﾞﾌﾛｰ仕様書
1902
1904
1905
ﾃﾞｰﾀﾍﾞｰｽ
解析情報
ﾌﾟﾛｸﾞﾗﾑ
解析情報
ｴﾝﾃｨﾃｨ
登録情報
DISP
ﾊﾟﾗﾒﾝﾄ
定義情報

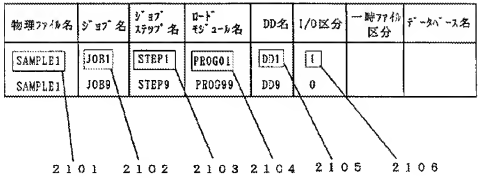
【図17】


開　始
101
ユティリティ情報登録
102
ＪＣＬ解析
103
プログラム解析
1101
データベース解析
JCL/ソース、JCL/DB
関連情報作成
1701
1702
関連情報を
編集するか
No
Yes
1703
JCL/ソース、JCL/DB
関連情報編集
③
JCL解析情報読込み
104
ジョブフロー基本情報作成
1104
②
1704
ロードモジュールに
関連する情報が
あるか
No
Yes
1705
関連情報をキーに
入出力区分を抽出
1706
ロードモジュール名分
全て抽出したか
No
Yes
①
①
111
全てのファイルの
入出力区分抽出を
したか
No
②
Yes
1105
ジョブフロー仕様書作成
113
次仕様書を作成
するか
No
③
Yes
1707
ファイル/ジョブ、ファイル/
プログラム情報作成
終　了
110
解析したJCLより
入出力区分抽出

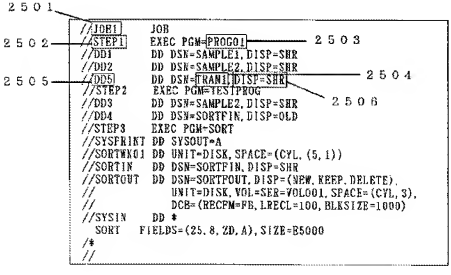
【図20】


JCL解析
情報
2001
2004
ｼﾞｮﾌﾞﾌﾛｰ仕様書
作成プログラム
ｼﾞｮﾌﾞﾌﾛｰ仕様書
2002
2003
ﾛｰﾄﾞﾓｼﾞｭｰﾙ名
ﾌﾟﾛｸﾞﾗﾑ名
ﾌﾟﾛｸﾞﾗﾑ区分
ﾃﾞｰﾀﾍﾞｰｽ名
TESTPROG
DISP
ｵﾍﾟﾗﾝﾄﾞ
定義情報

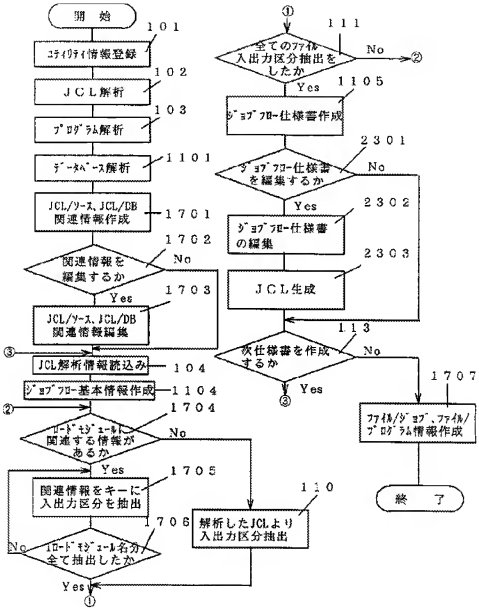
【図21】


物理ﾌｧｲﾙ名
ｼﾞｮﾌﾞ名
ｼﾞｮﾌﾞｽﾃｯﾌﾟ名
ﾛｰﾄﾞﾓｼﾞｭｰﾙ名
DD名
I/O区分
一時ﾌｧｲﾙ区分
ﾃﾞｰﾀﾍﾞｰｽ名
SAMPLE1
JOB1
STEP1
PROG01
DD1
I
SAMPLE1
JOB9
STEP9
PROG99
DD9
O
2101
2102
2103
2104
2105
2106

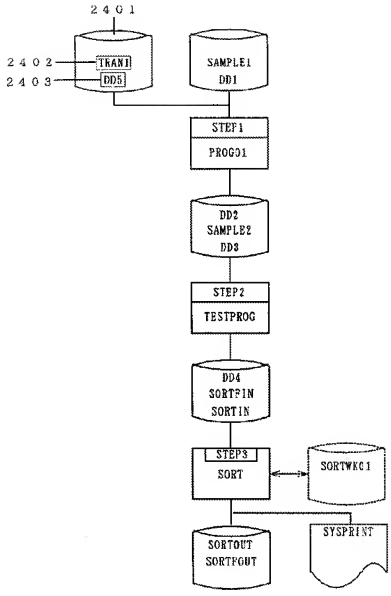
【図25】


2501
//JOB1 JOB
2502
//STEP1 EXEC PGM=PROG01
2503
//DD1 DD DSN=SAMPLE1,DISP=SHR
//DD2 DD DSN=SAMPLE2,DISP=SHR
2504
2505
//DD5 DD DSN=TRAN1,DISP=SHR
//STEP2 EXEC PGM=TESTPROC
2506
//DD3 DD DSN=SAMPLE2,DISP=SHR
//DD4 DD DSN=SORTFIN,DISP=OLD
//STEP3 EXEC SORT
//SYSPRINT DD SYSOUT=A
//SORTWK01 DD UNIT=DISK,SPACE=(CYL,(5,1))
//SORTIN DD DSN=SORTFIN,DISP=SHR
//SORTOUT DD DSN=SORTPOUT,DISP=(NEW,KEEP,DELETE),
// UNIT=DISK,VOL=SER=VOL001,SPACE=(CYL,3),
// DCB=(RECFM=FB,LRECL=100,BLKSIZE=1000)
//SYSIN DD *
SORT FIELDS=(25,8,ZD,A),SIZE=B5000
/*
//

【図２３】


開　始
１０１
ユティリティ情報登録
１０２
ＪＣＬ解析
１０３
プログラム解析
１１０１
データベース解析
１７０１
JCL/ソース、JCL/DB
関連情報作成
１７０２
関連情報を
編集するか
No
Yes
１７０３
JCL/ソース、JCL/DB
関連情報編集
③
１０４
JCL解析情報読込み
１１０４
ジョブフロー基本情報作成
②
１７０４
ロードモジュールに
関連する情報が
あるか
No
Yes
１７０５
関連情報をキーに
入出力区分を抽出
１７０６
ロードモジュール名分
全て抽出したか
No
Yes
①
①
１１１
全てのファイル
入出力区分抽出を
したか
No
②
Yes
１１０５
ジョブフロー仕様書作成
２３０１
ジョブフロー仕様書
を編集するか
No
Yes
２３０２
ジョブフロー仕様書
の編集
２３０３
ＪＣＬ生成
１１３
次仕様書を作成
するか
No
Yes
②
１７０７
ファイル/ジョブ、ファイル/
プログラム情報作成
１１０
解析したJCLより
入出力区分抽出
終　了

【図２４】



フロントページの続き

(72)発明者　内藤　一郎
神奈川県川崎市麻生区王禅寺1099番地　株式会社日立製作所システム開発研究所内

(72)発明者　月野　浩
東京都大田区大森北三丁目２番16号　日立システムエンジニアリング株式会社内